WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences | Cases

Search Results -

Terms	Documents
L12 and rows same extract$	3

Database: US Patents Full-Text Database
US Pre-Grant Publication Full-Text Database
JPO Abstracts Database
EPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search: Refine Search

Recall Text Clear

Search History

DATE: Wednesday, January 22, 2003 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=USPT,PGPB,JPAB,EPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L15	L12 and rows same extract$	3	L15
L14	L13 and table near recovery	5	L14
L13	L11 and (tablespace or table adj space or table-space)	361	L13
L12	L11 and table near recovery	41	L12
L11	database or data adj base	198181	L11
L10	(((707/204)!.CCLS.))	576	L10
L9	(((707/203)!.CCLS.))	753	L9
L8	(((707/202)!.CCLS.))	548	L8
L7	(((707/201)!.CCLS.))	722	L7
L6	(((707/200)!.CCLS.))	1143	L6
L5	(((707/102)!.CCLS.))	1473	L5
L4	(((707/101)!.CCLS.))	1010	L4
L3	(((707/100)!.CCLS.))	1384	L3
L2	(((707/7)!.CCLS.))	574	L2
L1	((707/1)!.CCLS.)	2104	L1

END OF SEARCH HISTORY

WEST

Generate Collection | Print

L14: Entry 4 of 5 File: USPT Jun 22, 1993

US-PAT-NO: 5222235
DOCUMENT-IDENTIFIER: US 5222235 A

TITLE: Databases system for permitting concurrent indexing and reloading of data by early simulating the reload process to determine final locations of the data

DATE-ISSUED: June 22, 1993

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hintz; Thomas E.	Austin	TX		
Cunningham; William R.	Austin	TX		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
BMC Software, Inc.	Sugar Land	TX			02

APPL-NO: 07/ 473663 [PALM]
DATE FILED: February 1, 1990

INT-CL: [05] G06F 3/00, G06F 7/00, G06F 9/38

US-CL-ISSUED: 395/600; 364/283.1, 364/974, 364/264.3, 364/DIG.1, 395/500, 395/800
US-CL-CURRENT: 707/101; 703/22

FIELD-OF-SEARCH: 395/500, 395/600, 395/800

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4247892	January 1981	Laurence et al.	364/200
☐	4514826	April 1985	Iwata et al.	364/900
☐	4575798	March 1986	Lindstrom et al.	364/300
☐	4644471	February 1987	Kojima et al.	395/600
☐	4679139	July 1987	Durbin	395/425
☐	4714992	December 1987	Gladney et al.	364/200
☐	4819156	May 1989	DeLorme et al.	364/200
☐	4831515	May 1989	Kamada et al.	364/200
☐	5053945	October 1991	Whisler	364/239.6
☐	5060147	October 1991	Mattheyses	395/650

ART-UNIT: 232

PRIMARY-EXAMINER: Lee; Thomas C.

ASSISTANT-EXAMINER: An; Meng-Ai T.

ABSTRACT:

The reorganization method of DB2 data files explores parallel processing, and asynchronous I/O to a great extent. It includes means to estimate an optimum configuration of system resources, such as storage devices (DASD devices), memory, and CPUs, etc, during reorganizations. The method mainly consists of four components, (1) concurrent indexing, (2) concurrent unloading of data file partitions, (3) efficient reloading of DB2 data pages and DB2 space maps, and (4) means to reduce access constraints to the DB2 recovery table.

1 Claims, 7 Drawing figures

L14: Entry 4 of 5 File: USPT Jun 22, 1993

DOCUMENT-IDENTIFIER: US 5222235 A
TITLE: Databases system for permitting concurrent indexing and reloading of data by early simulating the reload process to determine final locations of the data

Abstract Text (1):
The reorganization method of DB2 data files explores parallel processing, and asynchronous I/O to a great extent. It includes means to estimate an optimum configuration of system resources, such as storage devices (DASD devices), memory, and CPUs, etc, during reorganizations. The method mainly consists of four components, (1) concurrent indexing, (2) concurrent unloading of data file partitions, (3) efficient reloading of DB2 data pages and DB2 space maps, and (4) means to reduce access constraints to the DB2 recovery table.

Brief Summary Text (3):
1.1 Databases

Brief Summary Text (4):
1.2 Database Indexes

Brief Summary Text (5):
1.3 Database Disorganization

Brief Summary Text (18):
2.4 Updates of the Recovery Table

Brief Summary Text (41):
The present invention relates to a method for efficiently reorganizing databases that are stored in a computer system.

Brief Summary Text (42):
1.1 Databases

Brief Summary Text (43):
As illustrated in FIG. 1, a database file typically includes "records" that are divided into "fields". A record in a database file may be thought of as a row in a table, containing, for example, all the information for a client. Fields may be thought of as columns in a table. For example, one field in a database may correspond to the name of a client, another field of the same record may correspond to the client's company.

Brief Summary Text (44):
Fields commonly have specific sizes that are determined when the database file is constructed. For example, a field containing dates may have two character positions each for the day and for the month, and four character positions for the year (a character position corresponds to a letter of text, numerical digit or punctuations etc.). Variable size fields may also be implemented in the database. These fields may acquire additional character positions as required. For example, a field containing comments about a client may contain a short phrase or a complete paragraph of text.

Brief Summary Text (45):
Records in a database may be rearranged into a desired order, by sorting, whereas fields are fixed with respect to the record. Hence, the fields are common to the database, even though the field contents are not.

Brief Summary Text (46):
It is often desired to be able to retrieve information quickly from the database. If a database is randomly organized, in no specific order, it may be necessary to check all data, in "linear search" to find the desired information.

Brief Summary Text (47):
If the data in a database are organized on a specific field, for example, on the name of a client, it is necessary to search only a small portion of the database to find the desired client. This corresponds to the normal use of, e.g., a dictionary or telephone directory. One possible approach to perform a search would be to divide the directory into two equal halves, and then determine which half contains the desired information. The half that does not contain the information is excluded, and the remaining half is again divided into two equally sized portions. The process continues until the information is found. This process might seem slow, but an item in a list of a thousand items will be located with ten or fewer process steps. Many computers perform a searches as described above.

Brief Summary Text (48):
1.2 Database Indexes

Brief Summary Text (49):
Most databases contain "support" files, in addition to the data files (data files store the actual information that is of interest to the user or his clients etc.). The "support" files may include "index" files. An index file works like the index in a book; it contains a key (corresponds to a word or phrase in the index of the book) and a pointer (corresponds to the page number in the index of the book).

Brief Summary Text (51):
An index file typically contains "key/pointer" pairs that are related to a few fields in the database (i.e. a database may be arranged according to client names and telephone numbers). Each pointer is associated with a key, i.e., a copy of a data field from the original database. The pointer keeps track of where the associated record is stored in the data base. Naturally the index file has to be sorted when information is added or removed from the database, but the index file is much smaller than the database and is therefore quicker to sort. Furthermore, the organization of an index file may be based on a pointer structure (e.g., the well-known B-tree or B-plus tree organization).

Brief Summary Text (52):
1.3 Database Disorganization

Brief Summary Text (53):
The continual insertion, deletion, and updating of databases, IBM's DATABASE 2 (DB2) for mainframe computers in particular, causes data in the database to become disorganized. Disorganization can take the form of fragmented free space, data indirection, out-of-order data, and out-of-order index pages.

Brief Summary Text (60):
IBM developed a product, DB2 REORG, that can reorganize existing DB2 data and index files. The result is a database with reduced diorganization. However, no user access to the database is allowed during reorginization. Hence, it is usually very desirable to perform the reorganization at a minimum amount of time as well as to be able to predict the time required.

Brief Summary Text (61):
In addition, it is necessary to record that a reorganization has been performed for recovery purposes. Information about execution of a utility is kept in one of the database managers special catalog tables. These tables (in normal operation) cannot be updated directly by a user. The database manager maintains these tables.

Brief Summary Text (67):
As illustrated in FIG. 2, a DB2 defines its own environment where data structures and file I/O are unique. A database may contain several "table spaces" and "indexes". A table space is a group of tables, and a table is a collection of data records (a table is what is generally termed database). The database may be stored an a single storage device or on multiple storage devices (disk drive, tape etc.) depending on whether a table is "simple", or "partitioned", or "segmented".

Brief Summary Text (73):
A DB2 index is a set of pointers to data in a table. The index is stored separately from the table, but in the same database. The index is maintained by DB2, but can be checked, repaired, or reorganized by a user.

Brief Summary Text (75):
Each index occupies its own index space (similar to table spaces), which is created simultaneously with the index. If a clustering index is created for a table in a partitioned table space, the index will also be partitioned into multiple index spaces.

Brief Summary Text (78):
I/O is handled by DB2 in data units called "pages". A table space is divided into several equal sized portions (pages). DB2 reads and writes (I/O) data one page at a time.

Brief Summary Text (81):
DB2 uses several database tables to track information about user databases, and for its own internal operation. DB2 limits access to these tables, in order to prevent users from accidentally changing vital information.

Brief Summary Text (82):
One of DB2's internal tables tracks the "access path" to database tables. An access path is a set of descriptors of what databases a program access and is allowed to access. This table is updated by DB2 as users enter new applications and tables into the DB2 environment and prevents users from accidentally altering DB2 system files.

Brief Summary Text (83):
Other essential DB2 system tables are a set of "catalog tables", one of which is the "recovery table" (e.g., SYSIBM.SYSCOPY) which is the table file being modified. It maintains information necessary to recover damaged data. However, there are cases when the tables inside the catalog table cannot be modified.

Brief Summary Text (84):
When the prior art REORG program is run, it updates the recovery table to reflect that some tables have been reorganized. Hence, it is vital that a program or routine that reorganizes one or more tables in the DB2 environment be able to update the table. Reorganizing DB2 tables without updating the recovery table will not allow the affected tables to be recovered correctly in the event of damage.

Brief Summary Text (96):
The execution speed of the UNLOAD routine is highly dependent on the amount of I/O required (input or output operations). The amount of I/O is, in turn, dependent on the amount of disorganization of the database. Disorganization results in an increased amount of I/O operations and possibly very inefficient ordering of these operations.

Brief Summary Text (105):
The execution speed of the prior UNLOAD routine is highly dependent on the amount of I/O required (input or output operations). The amount of I/O is, in turn, a function of the amount of disorganization of the data base. Disorganization results in an increased amount of non-sequential I/O. Performance may be further degraded by using one of the indexes to unload data, depending on the amount of "out-of-orderness". If the index is severely disorganized, many additional non-sequential I/O cycles may be required to resolve indirect pointer references.

Brief Summary Text (113):
The improved reorgainization method comprises four major routines. The routines include, but are not limited to, (1) concurrent indexing and sorting (concurrent with reloading data), (2) concurrent unloading of partitions, (3) efficient treatment of space maps and data pages during reload, and (4) means to reduce access constraints to the recovery table.

Brief Summary Text (114):
These four major routines may not necessarily appear as isolated sub-programs or routines in the method. For example, indexing usually takes place while the data is reloaded into the database.

Brief Summary Text (117):
The heat of the problem lies in that pointers must be assigned before key/pointer pairs can be written to the index work file (because indexing entails assigning values from the data base to the key/pointer pairs that are stored in the index file). Furthermore, indexing requires that the key values and their locations in the database be known before indexing starts.

Brief Summary Text (124):
2.4 Updates of the Recovery Table

Brief Summary Text (125):
As noted above, the recovery table contains information about the data files and their tables and indexes. Access mechanisms within the method of the invention allow access to several DB2 system files, among them the recovery table.

Brief Summary Text (126):
By creating a table with an identical structure to the recovery table, DB2 permits access to this table because it is a user table. The program or routine that is to access the recovery table is then written as if it would access the copy instead. The program, or routine, is then compiled and built, utilizing standard programming tools familiar to those skilled in the art. The DB2 file containing the access paths is modified using the standard DB2 repair utility. The modification includes changing the database ID and the table ID of the duplicate files, to those used by the recovery table. DB2 is thereby "fooled" into believing that it is the user copy of the recovery table that is being accessed, when in reality the recovery table itself is being accessed.

Drawing Description Text (2):
FIG. 1 illustrates a typical database.

Drawing Description Text (3):
FIG. 2 illustrates a specific DB2 database.

Detailed Description Text (8):
The main program, Appendix A, illustrates the main events that take place during program execution. It directs the process to the UNLOAD and RELOAD WITH CONCURRENT INDEXING routines in sequence; initiates the production of image copies; and performs updates to the recovery table.

Detailed Description Text (15):
The keys that are used in the indexes are extracted from the database row, at step B5, and the row is stored until further processing can take place, at step B6. The key/pointer pair (key/RID pair) is temporarily stored until further processing can take place, at step B7, and the process is repeated by processing subsequent data rows, at step B7.

Detailed Description Text (19):
The next step is to determine if sorting of the database rows is to be performed during the actual unloading process, at step C3. The "order" is a parameter which is set prior to starting the reorganization process; it determines when the database rows are sorted. If this parameter is set then the MAXIMUM TASKS ROUTINE is performed. This routine determines the maximum number of tasks that can be available, depending on system resources available for the sort routine.

Detailed Description Text (77):
The process starts by initializing a space map, at step N1. The space map is written to a database dataset at page one (I/O page number one), at step N2. The page number of the first page is stored in the current space map and in a temporary variable, MAP#, at step N3. The first data page is initialized in memory, at step N4. A row (data record) is loaded into the current page in memory, at step N5. A test is performed, at step N6, to determine if the current page is full. If not, processing continues with performing the SPACE MAP SHIFT routine, at step N7.

CLAIMS:

1. A computer-implemented method of reorganizing a DB2 database comprising data organized into rows and stored in a row-data set, each said row having at least one index field containing an index-key value,

said computer-implemented method comprising the steps of:

(a) for each said row of data,

(1) retrieving said row of data from said row-data set; (2) obtaining a row identifier, referred to as a RID, for said row of data by simulating a process of storing said row of data to a table space;

(3) writing said row of data to an interim row-data set;

(4) for each said index field,

(A) determining the index-key value contained in the index field, and

(B) writing said index-key value for said row and the RID for said row to an interim index-data set corresponding to said index field;

(b) executing a plurality of processes in parallel to:

(1) retrieve said rows of data from said interim row-data set and write said rows of data to a row-data set; and

(2) retrieve said index-key values and said RIDs from said interim index-data set, sorting index-key value/RID pairs by index-key value, and writing an index-key value/RID pair to an index-data set.

L14: Entry 3 of 5 File: USPT Sep 12, 2000

DOCUMENT-IDENTIFIER: US 6119128 A
TITLE: Recovering different types of objects with one pass of the log

Abstract Text (1):
A method, apparatus, and article of manufacture for a computer implemented recovery system for restoring a database in a computer. The database contains objects and is stored on a primary data storage device connected to the computer. Objects of different types in the database are copied from the primary data storage device to a secondary data storage device. Modifications to the objects are logged in a log file. A recovery indicator is received that indicates that recovery of the objects in the database is required. The objects are copied from the secondary data storage device to the database on the primary data storage device. Modifications in the log file are applied to the copied objects during one pass through the log file.

Brief Summary Text (3):
This invention relates in general to computer-implemented database systems, and, in particular, to recovering different types of objects with one pass of the log.

Brief Summary Text (5):
Databases are computerized information storage and retrieval systems. A Relational Database Management System (RDBMS) is a database management system (DBMS) which uses relational techniques for storing and retrieving data. Relational databases are organized into tables which consist of rows and columns of data. The rows are formally called tuples. A database will typically have many tables and each table will typically have multiple tuples and multiple columns. The tables are typically stored on direct access storage devices (DASD), such as magnetic or optical disk drives for semipermanent storage.

Brief Summary Text (6):
A table is assigned to a tablespace. The tablespace contains one or more datasets. In this way, the data from a table is assigned to physical storage on DASD. Each tablespace is physically divided into equal units called pages. The size of the tablespace's pages is based on the page size of the bufferpool specified in the tablespace's creation statement. The bufferpool is an area of virtual storage that is used to store data temporarily. A tablespace can be partitioned, in which case a table may be divided among the tablespace's partitions, with each partition stored as a separate dataset. Partitions are typically used for very large tables.

Brief Summary Text (9):
Typically, the database containing partitions and indexes is stored on a data storage device, called a primary data storage device. The partitions are periodically copied to another data storage device, called a secondary data storage device, for recovery purposes. In particular, the partitions stored on the primary data storage device may be corrupted, for example, due to a system failure during a flood, or a user may want to remove modifications to the data (i.e., back out the changes). In either case, for recovery, the partitions are typically copied from the secondary data storage device to the primary data storage device. Next, using the log file, the copied data is modified based on the operations in the log file. Then, the indexes are rebuilt. In particular, to rebuild the indexes, keys are copied from each row of each partition, sorted, and then used to create a partitioning index. Additionally, the table index is rebuilt via the same technique.

Brief Summary Text (12):
To overcome the limitations in the prior art described above, and to overcome other limitations that will become apparent upon reading and understanding the present specification, the present invention discloses a method, apparatus, and article of manufacture for a computer implemented recovery system for restoring a database in a computer.

Brief Summary Text (13):

In accordance with the present invention, the database contains objects and is stored on a primary data storage device connected to the computer. Objects of different types in the database are copied from the primary data storage device to a secondary data storage device. Modifications to the objects are logged in a log file. A recovery indicator is received that indicates that recovery of the objects in the database is required. The objects are copied from the secondary data storage device to the database on the primary data storage device. Modifications in the log file are applied to the copied objects during one pass through the log file.

Brief Summary Text (14):
An object of the invention is to provide an improved recovery system for a database. Another object of the invention is to provide recovery for partitions, partitioning indexes, and table indexes simultaneously. Yet another object of the invention is to provide a recovery system for a database that requires only one pass of a log file to apply modifications to the database.

Drawing Description Text (4):
FIG. 2 illustrates a conventional system for recovery of a database;

Drawing Description Text (7):
FIG. 5 is a flow diagram illustrating the steps performed by the recovery system prior to recovery of a database in accordance with the present invention; and

Drawing Description Text (8):
FIG. 6 is a flow diagram illustrating the steps performed by the recovery system to recover a database in accordance with the present invention.

Detailed Description Text (4):
FIG. 1 illustrates an exemplary computer hardware environment that could be used in accordance with the present invention. In the exemplary environment, a computer system 102 is comprised of one or more processors connected to one or more data storage devices 104 and 106 that store one or more relational databases, such as a fixed or hard disk drive, a floppy disk drive, a CDROM drive, a tape drive, or other device.

Detailed Description Text (5):
Operators of the computer system 102 use a standard operator interface 108, such as IMS/DB/DC.RTM., CICS.RTM., TSO.RTM., OS/390.RTM. or other similar interface, to transmit electrical signals to and from the computer system 102 that represent commands for performing various search and retrieval functions, termed queries, against the databases. In the present invention, these queries conform to the Structured Query Language (SQL) standard, and invoke functions performed by Relational DataBase Management System (RDBMS) software. In the preferred embodiment of the present invention, the RDBMS software comprises the DB2.RTM. product offered by IBM for the MVS.RTM. or OS/390.RTM. operating systems. Those skilled in the art will recognize, however, that the present invention has application program to any RDBMS software that uses SQL.

Detailed Description Text (6):
As illustrated in FIG. 1, the DB2.RTM. architecture for the MVS.RTM. operating system includes three major components: the Internal Resource Lock Manager (IRLM) 110, the Systems Services module 112, and the Database Services module 1 14. The IRLM 110 handles locking services for the DB2.RTM. architecture, which treats data as a shared resource, thereby allowing any number of users to access the same data simultaneously. Thus concurrency control is required to isolate users and to maintain data integrity. The Systems Services module 112 controls the overall DB2.RTM. execution environment, including managing log data sets 106, gathering statistics, handling startup and shutdown, and providing management support.

Detailed Description Text (7):
At the center of the DB2.RTM. architecture is the Database Services module 114. The Database Services module 114 contains several submodules, including the Relational Database System (RDS) 116, the Data Manager 118, the Buffer Manager 120, the Recovery System 122, and other components 124, such as an SQL compiler/interpreter. These submodules support the functions of the SQL language, i.e. definition, access control, interpretation, compilation, database retrieval, and update of user and system data. The Recovery System 122 works with the components of the computer system 102 to restore a database.

Detailed Description Text (8):
The present invention is generally implemented using SQL statements executed under the control of the Database Services module 114. The Database Services module 114 retrieves or receives the SQL statements, wherein the SQL statements are generally

stored in a text file on [redacted]e data storage devices 104 and [redacted]6 or are interactively entered into the computer system 102 by an operator sitting at a monitor 124 via operator interface 108. The Database Services module 114 then derives or synthesizes instructions from the SQL statements for execution by the computer system 102.

Detailed Description Text (13):
The present invention provides a recovery system 122 for recovering different types of objects using only one pass through a log file. In particular, table partitions of a database, along with indexes (e.g., partitioning indexes and table indexes), are copied to one or more data storage devices, such as magnetic tape. The database may be stored on a primary data storage device, while the copies of the database partitions and indexes are stored on a secondary data storage device. The primary and secondary data storage devices could be the same or different devices.

Detailed Description Text (15):
modifications are logged in a log file. If recovery of the table partitions and partitioning indexes are required, the recovery system 122 of the present invention copies the table partitions and partitioning indexes from the secondary data storage device back to the database. Then, the recovery system 122 modifies both the table partitions and the partitioning indexes while making one pass through the log file. That is, the recovery system 122 extracts all of the pertinent log records containing updates to all of the objects being recovered in a single read pass of logged changes.

Detailed Description Text (16):
The recovery system 122 allows for independent recovery of the data and indexes, and a significant decrease in elapsed time since the log file updates are done for all objects in the database with one pass through the log file.

Detailed Description Text (17):
FIG. 2 illustrates a conventional system for recovery of a database. In a conventional system, partitions 200 and 202 of a database are copied from primary data storage devices to secondary data storage devices 204 and 206. In a conventional system, the partitioning indexes 208 and 210 and the table index 212 are not stored on secondary data storage devices. Then, when recovery is required, the conventional system copies the partitions 200 and 202 from the secondary data storage devices 204 and 206 to the database on the primary data storage devices. The conventional system applies modifications logged in a log file to the copied partitions. Then, the conventional system reads each row of each partition 200 and 202 and retrieves index keys 214 and 216 for each row of each partition 200 and 202. The index keys 214 and 216 are sorted and are used to rebuild indexes 208 and 210, respectively. Table index 212 is rebuilt in the same manner. This procedure has a high performance cost.

Detailed Description Text (18):
FIG. 3 illustrates the recovery system 122 in accordance with the present invention. Initially, the partitions 300 and 302 are copied to secondary data storage devices 304 and 306. Also, partitioning indexes 308 and 312 are copied to secondary data storage devices 310 and 314. The table index 316 is also copied to a secondary data storage device 318. Then, as application programs 320 modify the database by adding, updating, or deleting data via operations, the modifications are logged in the log file 322. The log file may be copied to a secondary data storage device 324 if the log file on the primary storage device becomes full. The log file 322 contains information identifying modifications to both the partitions and indexes.

Detailed Description Text (22):
Next, if the name of an employee is changed, the partition 400 and partitioning index 404 are modified. Then, the log file 410 contains an entry for "New Name, Old Name" that provides the new and old name of the employee whose name changed and an entry for "New Name Index, Old Name Index" that provides the index modification. Next, assuming that there is a loss of data, recovery of the data is required. Initially, the partition 400 and the partitioning index 404 are copied from secondary data storage devices back to the primary data storage device. Since, according to the copies file 416, these copies include all modifications up to range identifier L2, only operations after range identifier L2 are applied to the partition 400 and the partitioning index 404 to recover the database. Moreover, during one pass through the log file, the recovery system 122 identifies the required modifications and applies them to the partition 400 and the partitioning index 404.

Detailed Description Text (23):
FIG. 5 is a flow diagram illustrating the steps performed by the recovery system 122 prior to recovery of a database in accordance with the present invention. In Block

500, the recovery system [redacted] copies objects from a primary [redacted]ata storage device to a secondary data storage device. In Block 502, the recovery system 122 logs all operations for each object in the database that is modified. In Block 504, the recovery system 122 receives a recovery indicator.

Detailed Description Text (24):
FIG. 6 is a flow diagram illustrating the steps performed by the recovery system 122 to recover a database in accordance with the present invention. In Block 600, the recovery system 122 copies objects from the secondary data storage device to the primary data storage device. That is, each of the objects is replaced by an image copy taken at a previous time. The individual objects may be restored from the image copies concurrently with each other. In Block 602, the recovery system 122 determines the point in the log at which to start applying operations. In Block 604, the recovery system 122 applies log operations to all objects through one pass of the log file.

Detailed Description Text (28):
In summary, the present invention discloses a method, apparatus, and article of manufacture for a computer-implemented recovery system. The present invention provides an improved recovery system for a database. Additionally, the present invention provides recovery for partitions and partitioning indexes simultaneously. Moreover, the present invention provides a recovery system for a database that requires only one pass of a log file to apply modifications to the database.

Other Reference Publication (1):
"Incremental Data Base Log Image Copy", IBM Technical Disclosure Bulletin, vol. 25, No. 7B, pp. 3730-3732, 1982.

Other Reference Publication (3):
Fernando de Ferreira Rezende, et al., "Employing Object-Based LSNs in A Recovery Strategy", Database And Expert Systems Applications, 7th International Conference, DEXA '96, pp. 116-129, Sep. 9-13, 1996.

CLAIMS:

1. A method of restoring a database in a computer, the database containing objects and being stored on a primary data storage device connected to the computer, the method comprising the steps of:

copying objects of different types in the database from the primary data storage device to a secondary data storage device, wherein one of the objects is a table index for locating data in a table, and wherein one of the objects is a partitioning index for defining a scope of each partition and thereby assigning a row of the table to its respective partition;

logging modifications to the objects, including the table index and the partitioning index, in a log file;

receiving a recovery indicator indicating that recovery of the objects in the database is required;

copying the objects, including the table index and the partitioning index, from the secondary data storage device to the database on the primary data storage device; and

applying the modifications in the log file to the copied objects, including the table index and the partitioning index, during one pass through the log file.

9. An apparatus for restoring a database in a computer, comprising:

a computer having a primary data storage device connected thereto, wherein the primary data storage device stores a database containing objects;

one or more computer programs, performed by the computer, for copying objects of different types in the database from the primary data storage device to a secondary data storage device, wherein one of the objects is a table index for locating data in a table, and wherein one of the objects is a partitioning index for defining a scope of each partition and thereby assigning a row of the table to its respective partition, logging modifications to the objects, including the table index and the partitioning index, in a log file, receiving a recovery indicator indicating that recovery of the objects in the database is required, copying the objects, including the table index and the partitioning index, from the secondary data storage device to the database on the primary data storage device, and applying the modifications

in the log file to the copied objects, including the table index and the partitioning index, during one pass through the log file.

17. An article of manufacture comprising a computer program carrier readable by a computer and embodying one or more instructions executable by the computer to perform method steps for restoring a database, the database containing objects and being stored on a primary data storage device connected to the computer, the method comprising the steps of:

copying objects of different types in the database from the primary data storage device to a secondary data storage device, wherein one of the objects is a table index for locating data in a table, and wherein one of the objects is a partitioning index for defining a scope of each partition and thereby assigning a row of the table to its respective partition;

logging modifications to the objects, including the table index and the partitioning index, in a log file;

receiving a recovery indicator indicating that recovery of the objects in the database is required;

copying the objects, including the table index and the partitioning index, from the secondary data storage device to the database on the primary data storage device; and

applying the modifications in the log file to the copied objects, including the table index and the partitioning index, during one pass through the log file.

WEST

Generate Collection Print

L14: Entry 3 of 5 File: USPT Sep 12, 2000

US-PAT-NO: 6119128
DOCUMENT-IDENTIFIER: US 6119128 A

TITLE: Recovering different types of objects with one pass of the log

DATE-ISSUED: September 12, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Courter; Daniel Keith	Antioch	CA		
Hu; Ming-Hung	San Jose	CA		
Kunioka-Weis; Laura Michiko	Morgan Hill	CA		
Majithia; Thomas	San Jose	CA		
Matamoros; Deborah A.	San Jose	CA		
Ruddy; James Alan	Gilroy	CA		
Wang; Yufen	Saratoga	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY			02

APPL-NO: 09/ 050554 [PALM]
DATE FILED: March 30, 1998

INT-CL: [07] G06 F 12/00

US-CL-ISSUED: 707/202; 707/200, 707/201, 707/203, 707/204, 707/205
US-CL-CURRENT: 707/202; 707/200, 707/201, 707/203, 707/204, 707/205

FIELD-OF-SEARCH: 707/200, 707/201, 707/202, 707/203, 707/204, 707/205

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	4945474	July 1990	Elliott et al.	714/16
☐	5276872	January 1994	Lomet et al.	707/202
☐	5278982	January 1994	Daniels et al.	707/202
☐	5280611	January 1994	Mohan et al.	707/8
☐	5327532	July 1994	Ainsworth et al.	709/248
☐	5333303	July 1994	Mohan	714/20
☐	5455944	October 1995	Haderle et al.	707/202
☐	5455946	October 1995	Mohan et al.	707/202
☐	5561795	October 1996	Sarkar	707/202
☐	5561798	October 1996	Haderle et al.	707/202
☐	5574897	November 1996	Hermsmeier et al.	707/1
☐	5581750	December 1996	Haderle et al.	707/202
☐	5625820	April 1997	Hermsmeier et al.	707/202
☐	5721918	February 1998	Nilsoson	707/202
☐	5832508	November 1998	Sheman	707/200
☐	5873096	February 1999	Lim	707/201
☐	5903898	May 1999	Cohen	707/204
☐	5907848	May 1999	Zaiken	707/202
☐	5920873	July 1999	Van Huben	707/202
☐	5926816	July 1999	Bauer	707/8

OTHER PUBLICATIONS

"Incremental Data Base Log Image Copy", IBM Technical Disclosure Bulletin, vol. 25, No. 7B, pp. 3730-3732, 1982.
"Technique For Data Recovery excluding Portions of The Log Without Requiring a Full Image Copy", IBM Technical Disclosure Bulletin, vol. 36, No. 11, pp. 359-360, Nov. 1993.
Fernando de Ferreira Rezende, et al., "Employing Object-Based LSNs in A Recovery Strategy", Database And Expert Systems Applications, 7th International Conference, DEXA '96, pp. 116-129, Sep. 9-13, 1996.

ART-UNIT: 271

PRIMARY-EXAMINER: Black; Thomas G.

ASSISTANT-EXAMINER: Mizrahi; Diane D.

ABSTRACT:

A method, apparatus, and article of manufacture for a computer implemented recovery system for restoring a database in a computer. The database contains objects and is stored on a primary data storage device connected to the computer. Objects of different types in the database are copied from the primary data storage device to a secondary data storage device. Modifications to the objects are logged in a log file. A recovery indicator is received that indicates that recovery of the objects in the database is required. The objects are copied from the secondary data storage device to the database on the primary data storage device. Modifications in the log file are applied to the copied objects during one pass through the log file.

24 Claims, 6 Drawing figures

WEST

Generate Collection | Print

L15: Entry 1 of 3 File: USPT Feb 29, 2000

US-PAT-NO: 6032158
DOCUMENT-IDENTIFIER: US 6032158 A

TITLE: Apparatus and method for capturing and propagating changes from an operational database to data marts

DATE-ISSUED: February 29, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Mukhopadhyay; Pinaki	Sunnyvale	CA		
Nesamoney; Diaz	San Francisco	CA		
Sankaran; Mohan	Hayward	CA		
Suresh; Sankaran	Sunnyvale	CA		
Gupta; Sanjeev K.	Sunnyvale	CA		

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Informatica Corporation	Menlo Park	CA			02

APPL-NO: 08/ 850490 [PALM]
DATE FILED: May 2, 1997

INT-CL: [07] G06 F 17/30

US-CL-ISSUED: 707/201; 707/10
US-CL-CURRENT: 707/201; 707/10

FIELD-OF-SEARCH: 707/101, 707/201-204, 707/10

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
☐	5603024	February 1997	Goldring	707/203
☐	5675785	October 1997	Hall et al.	707/102
☐	5781911	July 1998	Young et al.	707/201
☐	5787415	July 1998	Jacobson et al.	707/2

OTHER PUBLICATIONS

Chan, R. "12 Steps of Creating a Successful Data Warehouse." Proceedings of the 8th International Database Workshop (Industrial Volume), Proceedings of the 8th International Hong Kong Computer Society Database Workshop. Data Mining, Data Warehousing and CLI, pp.227-248, XP002074804. ISBN 981-3083-53-0, 1997, Singapore, Springer-Verlag Singapore, Singapore.
Anand, V. J. et al. "Data Warehouse Architecture for DDS Applications." Australian Journal of Information Systems. Sep. 1996, AJIS Publishing, Australia, vol. 4, No. 1, pp. 43-53, XP002074805 ISSN 10399-7841.
Weyman, P. J. "The Case for a Process-Driven Approach to Data Warehousing." Database

and Network Journal, Feb[redacted]997, A. P. Publications, UK, vo[redacted] 27, No. 1, pp. 3-6, XP002074806, ISSN 0265-4490.
Mumick I. S. et al. "Maintenance of Data Cubes and Summary Tables in a Warehouse." SIGMOD 1997. ACM SIGMOD International Conference on Management of Data, Tucson, AZ, USA, 13-15, May 1997, vol. 26, No. 2, pp. 100-111, XP002074807, ISSN 0163-5808, SIGMOD Record, Jun. 1997, ACM, USA.
Curley, K. et al. "The Rationale for Developing a Corporate Data Warehouse Environment." OOIS '95. 1995 International Conference on Object Oriented Information Systems Proceedings, Proceedings of 1995, International Conference on Object Oriented Infromation Systems, Dublin, Ireland, Dec. 18-20, 1995, pp. 351-366, XP002074808, ISBN 3-540-76010-5, 1996, Berlin Gemany, Springer-Verlag, Germany.

ART-UNIT: 277

PRIMARY-EXAMINER: Vonbuhr; Maria N.

ABSTRACT:

A method for updating a target table of a data mart in response to changes made by a transaction to data stored in a source table of an operational database. Data that was changed in the source table by the transaction is stored in a dynamic image table of a change capture database. Data that was not changed in the source table by the transaction, but which is nevertheless required to be mapped to the target table, is stored in a static image table of the change capture database. The change capture database also contains relevant information regarding the transaction. Once the dynamic and static image tables are properly staged, the changes are propagated from the change capture database to the target tables of the data marts. In other words, data is extracted from the change capture database and subsequently transformed and loaded, thereby minimizing the impact to the operational database. Thereupon, the tables of the change capture database are truncated to discard data which is now no longer needed.

18 Claims, 8 Drawing figures